Proceeding of the IEEE International Conference on Robotics and Automation, San Francisco, CA, April 24-27, 2000.

4-D/RCS Reference Model Architecture for Unmanned Ground Vehicles

James S. Albus

Intelligent Systems Division
National Institute of Standards and Technology
Gaithersburg, MD 20899

Abstract

4-D/RCS is the reference model architecture currently being developed for the Demo III Experimental Unmanned Vehicle program. 4-D/RCS integrates the NIST (National Institute of Standards and Technology) RCS (Real-time Control System) with the German (Universitat der Bundeswehr Munchen) VaMoRs 4-D approach (3-Dimensions + time) to dynamic machine vision. The 4-D/RCS architecture consists of a hierarchy of computational nodes each of which contains behavior generation (BG), world modeling (WM), sensory processing (SP), and value judgment (VJ) processes. Each node also contains a knowledge database (KD) and an operator interface. These computational nodes are arranged such that the BG processes represent organizational units within a command and control hierarchy.

1 Introduction

4-D/RCS is the reference model architecture selected for the Demo III Experimental Unmanned Ground Vehicle (XUV). [1] 4-D/RCS integrates the NIST (National Institute of Standards and Technology) RCS (Real-time Control System) with the German (Universitat der Bundeswehr Munchen) VaMoRs 4-D approach [2] to dynamic machine vision. The 4-D/RCS architecture consists of a hierarchy of organizational units each of which has a supervisor agent and a set of subordinate agents that supervise units at the next lower level. Each node contains behavior generation (BG), world modeling (WM), sensory processing (SP), and value judgment (VJ) processes, and a knowledge database (KD). [3] A typical node is illustrated in Figure 1.

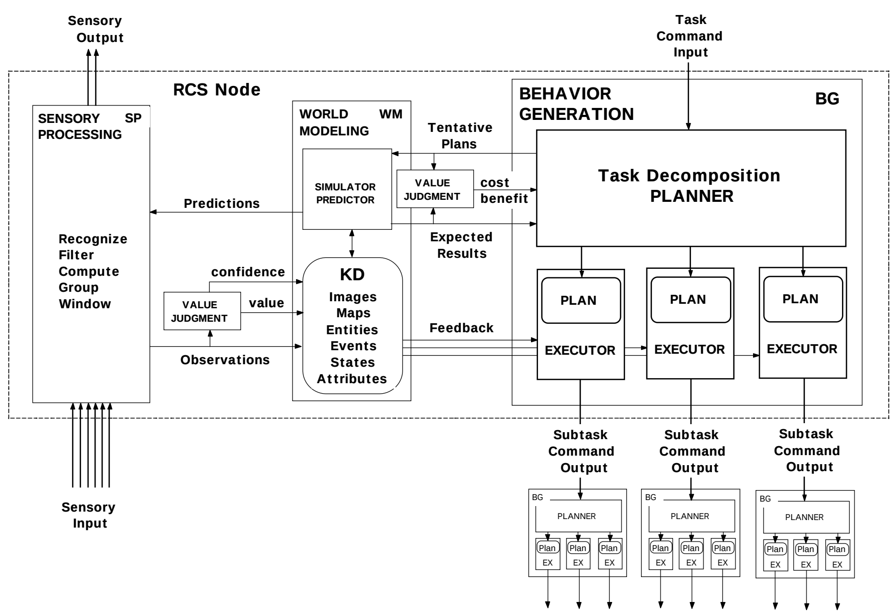

Figure 1. A typical 4-D/RCS computational node. Task command inputs come from a higher level behavior generation (BG) process in the 4-D/RCS hierarchy. Each input task command is decomposed into a plan consisting subtasks for subordinate BG processes. A world modeling (WM) process maintains a knowledge database (KD) that is the BG unit's best estimate of the external world. A sensory processing (SP) system operates on input from sensors by focusing attention (i.e., windowing), grouping, computing attributes, filtering, and recognizing entities, events, and situations. A value judgment (VJ) process evaluates expected results of tentative plans. A VJ process also evaluates entities, events, and situations entered into the KD.

Each BG process includes a planner module that accepts task command inputs from its supervisor and generates coordinated plans for subordinate BG processes. The BG planner hypothesizes tentative plans, WM predicts the probable results, and VJ evaluates the results of each tentative plan. The BG planner then selects the tentative plan with the best evaluation to be placed in the plan buffers in the BG Executors. There is an Executor that services each subordinate BG unit, issuing subtask commands, monitoring progress, compensating for errors and differences between planned and observed situations in the world, and reacting quickly to emergency conditions with appropriate actions. Feedback from a real-time knowledge database KD enables the executors to generate reactive behavior. SP and WM processes update the KD with images, maps, entities, events, attributes, and states necessary for both deliberative and reactive behavior. Coordination between subordinate BG processes is achieved by cross-coupling among plans and sharing of information among Executors via the KD. A review of projects that have used RCS is contained in [4]. A review that describes how RCS relates to other intelligent system architectures is contained in [5].

2 Structure

4-D/RCS computational nodes are arranged such that the BG processes represent organizational units within a command and control hierarchy. The BG command and control hierarchy for the first five levels of the Demo III Experimental Unmanned Vehicle (XUV) is shown in Figure 2.

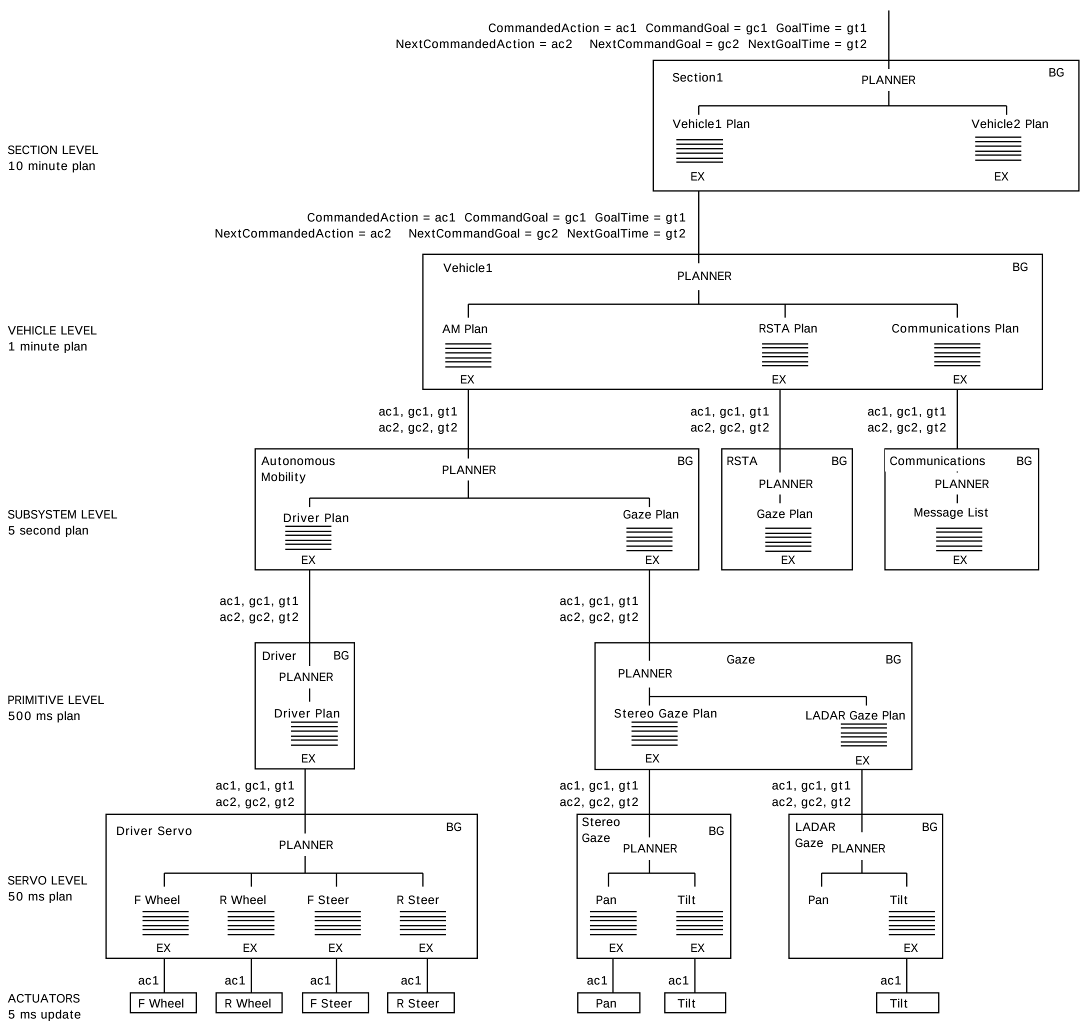

Figure 2. The command and plan structure for Demo III In the BG modules at each level there is a planner that produces one or more Plans for one or more subordinate BG modules. There is an Executor for each plan that communicates with the subordinate about how to integrate the lower level plan into the higher level plan.

The levels in the hierarchy are chosen based on time and spatial range and resolution. We have arbitrarily chosen factor of 10 as the interval between levels. This choice is on the high side of George Miller's "magic number 7, plus or minus 2." Miller's magic number was based on empirical estimates of the chunking between hierarchical levels in the human brain. [6]

The BG modules in Figure 2 are serviced by WM, KD, VJ, and SP modules at each level of the hierarchy. The KD at each level consists of maps, images, entities, events relationships, and state variables. Maps contain multiple overlays of labeled pixels that indicate the presence of entities and attributes such as obstacles, open areas, regions with cover, buildings, roads, trees, streams, fences, terrain elevation, surface roughness, and landmarks. Images contain attributes of pixels and segmented regions. Pixels and regions may have pointers to entity frames with class names and class attributes. Entities may be represented iconically by regions in images or maps, or symbolically by lists of attributes in frames. Temporal sequences may be grouped into events with pointers to event frames that may have class names and class attributes. Relationships and situations may be defined by semantic networks that link entities and events.

The KD is grounded at each level by SP through recursive estimation processes of hypothesize and test. SP directs attention, applies gestalt grouping hypotheses, correlates computed group attributes with predicted group attributes from the WM, and computes variance between computed and predicted attributes. Correlation offsets are used by the WM to estimate entity state variables such as position and velocity. Variance is used by the WM to update estimated values in the KD and to assess confidence in KD estimates.

VJ uses map overlays to compute cost and risk of traversing various regions on the map. At each level, a BG receives commands to perform actions and achieve goals. The BG planner hypothesizes various possible plans to achieve its commanded goal. The VJ computes the cost and risk of each hypothesized plan, and returns its evaluation to the planner. The planner selects the best plan to be sent to a set of executors that issue commands to subordinate subsystems.

Figure 3 shows the interactions between SP, WM, VJ, KD, and BG processes in the mobility chain of command for the first five levels in the Demo III architecture. On the right, Behavior Generation modules decompose high-level mission commands into low-level actions. The text beside the Planner and Executor at each level indicates the planning horizon, replanning rate, and reaction latency of commands at each level. Each planner has a world model simulator that is appropriate for the problems encountered at its level. In the center, each map as a range and resolution that is appropriate for path planning at its level. At each level, there are symbolic data structures and segmented images with labeled regions that describe entities, events, and situations that are relevant to decisions that must be made at that level. On the left is a sensory processing hierarchy that extracts information from the sensory data stream that is needed to keep the world model knowledge database current and accurate.

At the bottom are actuators that act on the world and sensors that measure phenomena in the world. The Demo III vehicles have a variety of sensors that include a laser range imager (LADAR), stereo CCD (charge coupled device) cameras, stereo forward looking infra red (FLIR) devices, a color CCD, a vegetation penetrating radar, GPS (Global Positioning System), an inertial navigation package, actuator feedback sensors, and a variety if internal sensors for measuring parameters such as engine temperature, speed, vibration, oil pressure, and fuel level. The vehicle also carries a Reconnaissance, Surveillance, and Target Acquisition (RSTA) mission package that includes long-range cameras and FLIRs, a laser range finder, and an acoustic array.

In Figure 3, the bottom (Servo) level has no map representation. The Servo level deals with actuator dynamics and reacts to sensory feedback from actuator sensors. The Primitive level map has range of 5 meters with resolution of 4 centimeters. This enables the vehicle to make small path corrections to avoid bumps and ruts during the 500 millisecond planning horizon of the Primitive level. The Primitive level also uses accelerometer data to compensate for vehicle dynamics. The Subsystem level map has range of 50 meters with resolution of 40 centimeters. This map is used to plan about 5 seconds into the future to find a path that avoids obstacles and provides a smooth and efficient ride. The Vehicle level map has a range of 500 meters with resolution of 4 meters. This map is used to plan paths about one minute into the future taking into account terrain features such as roads, bushes, gullies, or tree lines. At the Vehicle level and above, data from sensors is merged with a priori maps from military sources. The Section level map has a range of 5000 meters with resolution of 40 meters. This map is used to plan about 10 minutes into the future to accomplish tactical behaviors. Maps at platoon and battalion levels (not shown in Figure 3) are used to plan missions lasting about 2 and 24 hours respectively. These higher level maps are derived from standard sources of military maps. The choice of range and resolution of the maps was based on a preliminary design speed of 36 km per hour. For higher speeds, the range of the maps will be extended.

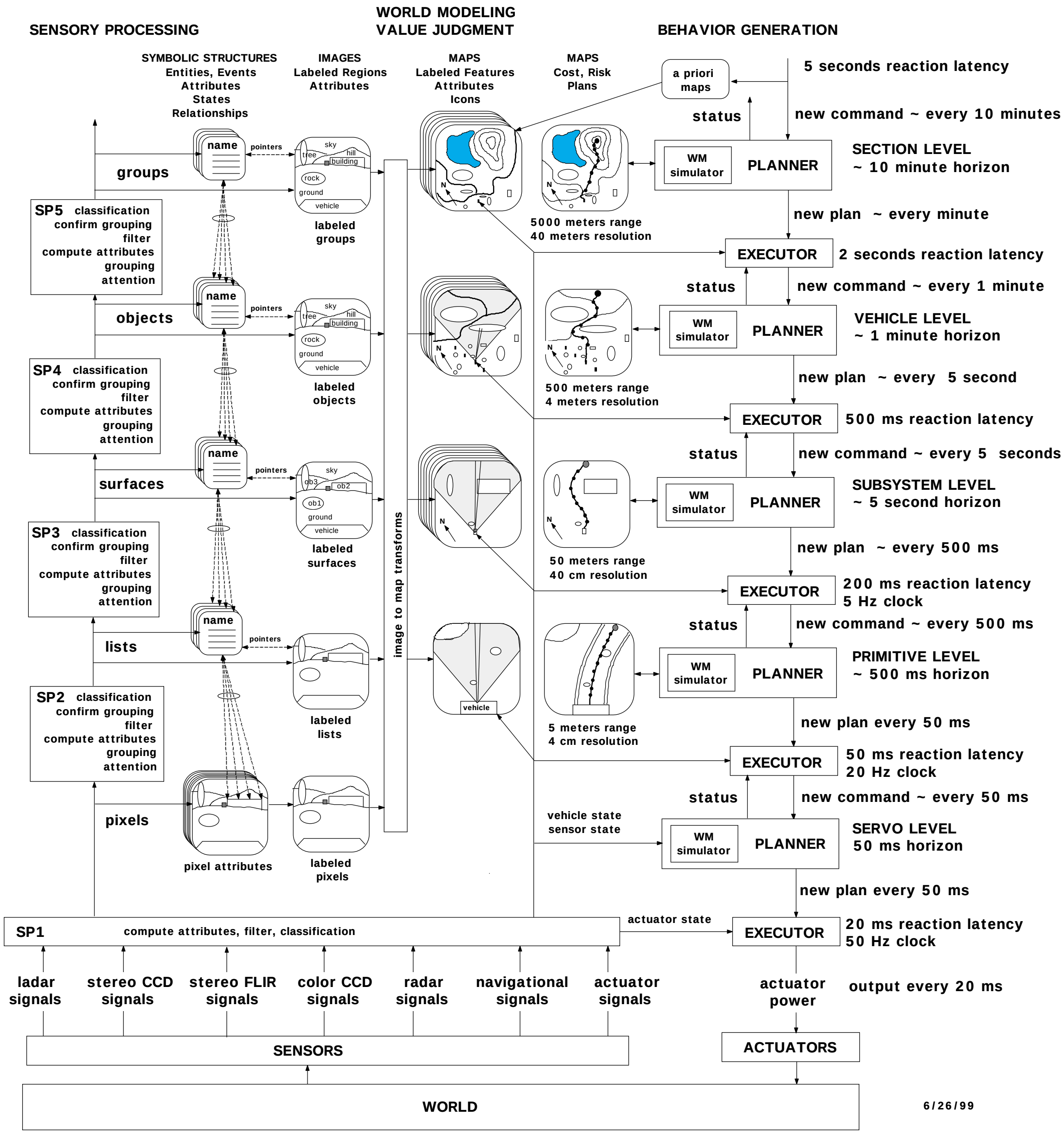

Figure 3. Five levels of the 4-D/RCS architecture. On the right are Planner and Executor modules. In the middle are maps for representing terrain features, road, bridges, vehicles, friendly/enemy positions, and the cost and risk of traversing various regions. On the left are Sensory Processing functions, symbolic representations of entities and events, and segmented images with labeled regions.

3 Commands

Commands into each BG module consist of six elements:

1) CommandedAction (ac1) describes the action to be performed and may include a set of modifiers such as priorities, mode, path constraints, acceptable cost, and required conditions.

2) CommandGoal (gc1) describes the desired state (or goal state) to be achieved by the action. Mobility system state typically includes the position, heading, velocity, and turning rate of the system being controlled. The goal may include the name of a target or object that is to be acted upon. It also may include a set of modifiers such as tolerance.

3) GoalTime (gt1) - defines the timing constraint on achieving the goal plus modifiers such as tolerance.

4) NextCommandedAction (ac2) - describes the planned next action to be performed plus modifiers.

5) NextCommandGoal (gc2) - describes the planned next goal state to be achieved plus modifiers.

6) NextGoalTime (gt2) - describes the timing constraint on achieving the next goal plus modifiers.

The planner in each BG process decomposes commands into plans for each of its subordinate BG processes. Each plan is designed to have about ten steps. For each plan, an Executor cycles through the plan issuing commands, monitoring progress, compensating for errors, and reacting to surprises and emergencies. For example, a command into the Vehicle level (4) for the first vehicle in a scout Section would have the form:

CommandedAction = $ac1^4_1$ CommandGoal = $gc1^4_1$ GoalTime = $gt1^4_1$ ~ t + 1 min
NCAction = $ac2^4_1$ NCGoal = $gc2^4_1$ NextGoalTime = $gt1^4_1$ ~ t + 2 min

This command would be decomposed into three plans for the Subsystem level of the form.

Autonomous Mobility Plan	RSTA Plan	Communications Plan
$ap1^3_1$, $gp1^3_1$, $gt1^3_1$~ t+5 sec	$ap1^3_2$, $gp1^3_2$, $gt1^3_2$	$ap1^3_3$, $gp1^3_3$, $gt1^3_3$
$ap2^3_1$, $gp2^3_1$, $gt2^3_1$~ t+10 sec	$ap2^3_2$, $gp2^3_2$, $gt2^3_2$	$ap2^3_3$, $gp2^3_3$, $gt2^3_3$
$ap3^3_1$, $gp3^3_1$, $gt3^3_1$~ t+15 sec	$ap3^3_2$, $gp3^3_2$, $gt3^3_2$	$ap3^3_3$, $gp3^3_3$, $gt3^3_3$
$ap4^3_1$, $gp4^3_1$, $gt4^3_1$~ t+20 sec	$ap4^3_2$, $gp4^3_2$, $gt4^3_2$	$ap4^3_3$, $gp4^3_3$, $gt4^3_3$
$ap5^3_1$, $gp5^3_1$, $gt5^3_1$~ t+25 sec	$ap5^3_2$, $gp5^3_2$, $gt5^3_2$	$ap5^3_3$, $gp5^3_3$, $gt5^3_3$
$ap6^3_1$, $gp6^3_1$, $gt6^3_1$~ t+30 sec	$ap6^3_2$, $gp6^3_2$, $gt6^3_2$	$ap6^3_3$, $gp6^3_3$, $gt6^3_3$
$ap7^3_1$, $gp7^3_1$, $gt7^3_1$~ t+35 sec	$ap7^3_2$, $gp7^3_2$, $gt7^3_2$	$ap7^3_3$, $gp7^3_3$, $gt7^3_3$
$ap8^3_1$, $gp8^3_1$, $gt8^3_1$~ t+40 sec	$ap8^3_2$, $gp8^3_2$, $gt8^3_2$	$ap8^3_3$, $gp8^3_3$, $gt8^3_3$
$ap9^3_1$, $gp9^3_1$, $gt9^3_1$~ t+50 sec	$ap9^3_2$, $gp9^3_2$, $gt9^3_2$	$ap9^3_3$, $gp9^3_3$, $gt9^3_3$
$ap10^3_1$, $gp10^3_1$, $gt10^3_1$~ t+1 min	$ap10^3_2$, $gp10^3_2$, $gt10^3_2$	$ap10^3_3$, $gp10^3_3$, $gt10^3_3$

where ap is action planned, gp is goal planned, and gt is planned goal time
and api^j_k is the i-th planned action for the k-th subordinate BG module at the j-th level

The Vehicle level Executor for the Autonomous Mobility (AM) Subsystem would then transform the first and second steps in the AM plan into a command to the AM BG module at the Subsystem level (3).

The 4-D/RCS architecture is designed to provide real-time intelligent control in a highly dynamic unstructured environment. It has seven hierarchical levels to deal with seven orders of magnitude difference between the space and time domains from the highest to the lowest level. The highest level planning horizon extends 24 hours into the future. The lowest level feedback loop cycles every 5 milliseconds. (Only five levels are shown in Figure 3.)

4 Replanning

Multiple levels of deliberative planning ensure that plans can be recomputed frequently enough that they never become obsolete. Planners generate new plans well before current plans are fully executed. Typically, replanning is completed by the time the first subgoal is achieved in the current plan (e.g., replanning at level 3 occurs about every 500 milliseconds.) Executors react to sensory feedback even faster[1] (e.g., reaction at level 3 occurs within 100 milliseconds.)

To achieve this rate of replanning, it is necessary to limit the amount of data in the world model that needs to be refreshed between each planning cycle. Multilevel representation of space limits the number of resolution elements required in maps and the amount of detail in symbolic data structures at each level. Multilevel representation of time limits the number of events and temporal detail required at each level. The world model in any node is rich and detailed within the region of attention, but contains only the amount of resolution in space and time required for making decisions in that node. This enables the world model in each node to be updated in real-time.

To replan frequently, it is also necessary to limit the amount of search required to generate new plans.

[1] Except at level 1 where replanning and reaction times are the same. At levels 2 and above, the difference between replanning and reacting becomes more significant with each successively higher level.

There are several ways to limit the search. One is to pre-compute and store plans that can be selected by a rule-based planner in response to the recognition of an object, event, or situation. A second approach is to limit the range and resolution of the state space that needs to be searched and evaluated. At each level, the range and resolution of maps can be limited to less than 64,000 resolution elements.

The 4-D/RCS architecture has an interface between deliberative and reactive execution in every node at every hierarchical level. This enables 4-D/RCS to fully realize the desirable traits of both deliberative and reactive control in a practical system. Multiple levels of deliberative planning ensure that plans can be recomputed frequently enough that they never become obsolete. Multiple levels of representation cause the planning search space to be limited in range and resolution, and plans to be limited in the number of steps and amount of detail. Multiple levels of feedback from the environment ensure that reactive behavior can be generated with a minimum of feedback time delay.

4 Disclaimer

It should be noted that only parts of the 4-D/RCS architecture have been implemented as of this date. The most complete implementation is in the BG hierarchy. BG modules have been implemented at the Servo, Primitive, Subsystem, and Vehicle levels. WM maps have been implemented at the Subsystem and Vehicle levels, and SP functions have been implemented at the pixel level. All regions detected as obstacles, clear space, or regions providing cover are represented only at the pixel level in LADAR and stereo images. This relatively primitive implementation has enabled the NIST HMMWV (Highly Mobile Multipurpose Wheeled Vehicle) to drive cross-country in mostly open terrain avoiding obstacles such as trees and fences at speeds of more than 25 km per hour.

The 4-D approach developed by the Universitat der Bundeswehr Munchen (UBM) for the 'VaMoRs-P' system [2] includes Road Detection and Tracking (RDT) and the Obstacle Detection and Tracking (ODT) algorithms. The 4-D approach uses recursive estimation methods to compute state variables for a set of objects in a 4-D representation (3-D space + time) of the world. This 4-D world model uses relatively simple object models and object recognition algorithms. It functions primarily in on-road situations. However, it runs in real-time at 25 Hz. The RDT and ODT systems have not yet been integrated into the 4-D/RCS architecture. However, the stand-alone results have been impressive. On-road driving by the UBM vehicle using the 4-D approach has achieved speeds up to 150 km per hour on the Autobaun in traffic with automatic lane changing. [2]

It is anticipated that all of the features described in this paper will be operational by the end of the Demo III program in the year 2001. We are confident that the 4-D/RCS architecture is sufficiently general and powerful and provides a sufficiently rich representation of the world to successfully perform all of the navigation, driving, and tactical behavior requirements of the Demo III program.

Acknowledgement

The Demo III program is sponsored by the Office of the Secretary of Defense and managed by the Army Research Laboratory, Human Engineering Laboratory. Chuck Shoemaker is the program manager.

References

[1] J. S. Albus, "4-D/RCS: A Reference Model Architecture for Demo III, Version 1.0," *NISTIR 5994*, National Institute of Standards and Technology, Gaithersburg, MD, March, 1997.

[2] E. D. Dickmanns, et. al., "The Seeing Passenger Car 'VaMoRs-P'," *International Symposium on Intelligent Vehicles'94,* Paris, October 24-26, 1994.

[3] J. S. Albus and A. M. Meystel, "A Reference Model Architecture for Design and Implementation of Intelligent Control in Large and Complex Systems," *International Journal of Intelligent Control and Systems,* Vol.1, No. 1, pp. 15-30, 1996

[4] J. S. Albus, "The NIST Real-time Control System (RCS): An Application Survey," *Proceedings of the AAAI 1995 Spring Symposium Series*, Stanford University, Menlo Park, CA, March 27-29, 1995.

[5] R. C. Arkin, *Behavior Based Robotics*, MIT Press, Cambridge, MA 1998

[6] G. A. Miller, "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information", *The Psychological Review*, 63, pp.71-97, 1956.